

# A Quantum Algorithm for Testing Juntas in Boolean Functions

Khaled El-Wazan <sup>a 1</sup>, Ahmed Younes <sup>b 1,2</sup>, and S. B. Doma <sup>c 1</sup>

<sup>1</sup>Department of Mathematics and Computer Science, Faculty of Science, Alexandria University, Egypt

<sup>2</sup>School of Computer Science, University of Birmingham, Birmingham, B15 2TT, United Kingdom

## Abstract

Given a Boolean function  $f$  with  $n$  variables provided as a black-box, this paper will propose a quantum algorithm for testing if a certain variable is junta or  $\epsilon$ -far from being junta. The proposed algorithm constructs another black-box using two copies of the provided black-box. The constructed black-box is used with the partial diffusion operator in an amplitude amplification technique to test whether the variable in question is junta or not using  $O(\sqrt{2^n})$  queries to the constructed black-box. The proposed algorithm considers a Boolean function of general form on contrary to relevant algorithms proposed by others.

## 1 Introduction

A  $k$ -junta is a function of  $n$  input variables which depends on at most  $k$  input variables out of the  $n$  input variables for this function, where  $k \leq n$ . The problem of finding whether a Boolean function is a  $k$ -junta function or  $\epsilon$ -far from being a  $k$ -junta function is, for example, considered a typical problem in machine learning in which there is often no way to explicitly discriminate relevant features to the learning process of an unknown function from other irrelevant features [1, 2]. Therefore, it is necessary useful to use an adequate algorithm for testing whether an unknown function is a  $k$ -junta function or not, before engaging in running any  $k$ -junta learning algorithm.

Quantum computers are probabilistic devices, which promise to do some types of computations more powerfully than classical computers. Many quantum algorithms have been presented, for example, P. Shor presented a quantum

---

<sup>a</sup>khaled.elwazan@alex-sci.edu.eg

<sup>b</sup>ayounes@alexu.edu.eg

<sup>c</sup>sbdoma@yahoo.com

algorithm for factorizing a composite integer into its prime factors in polynomial time [3]. L. Grover presented an algorithm for searching unstructured list of  $N$  items with quadratic speed-up over algorithms running on classical computers [4].

Bernstein-Vazirani's algorithm was one of the earliest quantum algorithms [7], which dealt with the oracle identification problem, where it is required to identify an unknown linear Boolean function given as a black-box. The complexity of this problem is measured by how many queries are needed to know the exact form of the function itself, which is  $O(n)$  on a classical computer, however, it requires a single query to the oracle using this quantum algorithm.

In 2007, Atici and Servado introduced a quantum  $\delta$ -property tester for  $k$ -junta Boolean functions using  $O(k/\delta)$  quantum queries and based on Fourier sampling [8]. As well, they introduced an algorithm for learning a  $k$ -junta to accuracy  $\delta$  that uses  $O(\delta^{-1}k \log k)$  quantum examples and  $O(2^k \log \delta^{-1})$  random examples.

Floess introduced a Bernstein-Vazirani based algorithm for finding which input variables the Boolean function depend on [9]. Amplitude amplification algorithm [4] is then used to increase the success probability of finding those variables which the function depend on, and runs in  $O(2^n)$ . As well, Floess proposed quantum algorithms for learning which variable resides in a linear, quadratic or cubic terms in the function only with the assumption that each variable appears in at most one term.

Hongwei presented a quantum algorithm which evaluates the influence of a variable on a Boolean function, using  $O(1)$  steps of Bernstein-Vazirani circuit [10]. Hongwei also discussed a probabilistic algorithm for learning quadratic and cubic functions of simple forms.

The aim of this paper is to propose an algorithm to test whether a black-box Boolean function  $f$  with  $n$  input variables is a  $k$ -junta function or  $\epsilon$ -far from being junta. The proposed algorithm can identify whether a variable  $x_i$  in the function  $f$  is relevant or not, using a new function  $g$  which is constructed from the given black-box Boolean function  $f$ . The new constructed function  $g$  is, then, used with an amplitude amplification algorithm, based on partial diffusion operator, to increase the success probability. As well, the algorithm works on any class of Boolean functions with success probability at least  $2/3$ .

The paper is organized as follows: Section 2 depicts a quantum search algorithm with more reliable behavior for both known and unknown number of matches. Section 3 introduces the construction of a new function  $g$  using the original black-box Boolean function  $f$  which will facilitate the junta property testing. Section 4 presents the proposed algorithm. Section 5 describes the proposed algorithm performance when testing any Boolean function regardless of its form. Section 6 compares the proposed algorithm with other relevant algorithms, followed by a general conclusion in sect. 7.

## 2 Quantum Search Algorithm

Let's consider having a list  $L$  of  $N = 2^n$  items, that has an oracle  $U_f$  which is used to access those items given that each item  $i$  is labeled with an integer  $\{0, 1, \dots, N-1\}$  and mapped to either 0 or 1 according to any certain property satisfied by  $i$ , *i.e.*  $f : L \rightarrow \{0, 1\}$ . The search problem is to find  $i \in L$  such that  $f(i) = 1$ .

L. Grover introduced in 1996 a novel approach for solving this typical problem [4] with quadratic speed-up over classical algorithms. The algorithm he proposed exploits quantum parallelism by preparing a uniform superposition which represents all the possible  $N$  items, starts marking the solutions using phase shift of  $-1$  using the oracle  $U_f$  and then amplifies the amplitudes of the solutions using inversion about the mean (diffusion operator). Grover's algorithm has shown to be optimal with high success probability if there is exactly one item  $i$  in the list  $L$  that satisfies the oracle  $U_f$ , and the algorithm requires approximately  $\pi/4\sqrt{N}$  iterations for that particular case [11]. The algorithm has been generalized in the case of known matches  $M$  that satisfies the oracle  $U_f$ , *i.e.*  $\forall j, \text{ for which } 1 \leq j \leq M \leq 3N/4, f(i_j) = 1$ , to require a number of  $\pi/4\sqrt{N/M}$  iterations. However, in the case of unknown number of matches, an algorithm is presented to find the number of matches and can be employed to decide the value of  $M$  [12]. It is found that this algorithm fails in the case of  $M > 3N/4$ .

Younes *et al.* [6] introduced a more reliable algorithm in the case of multiple matches than Grover's algorithm, and for fewer matches the algorithm runs in quadratic speed up similar to Grover's algorithm.

In the following section, Younes *et al.*'s algorithm for both known and unknown number of matches  $M$  will be reviewed since it will be used in our proposed algorithm.

### 2.1 In Case of Known Number of Matches $M$

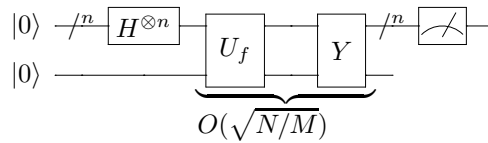


Figure 1: Quantum circuit for the quantum search algorithm [6].

For a list  $L$  of size  $N = 2^n$ , the steps are as follows:

1. Prepare a quantum register with  $n + 1$  qubits in a uniform superposition

$$|\varphi\rangle = \frac{1}{\sqrt{N}} \sum_{i=0}^{N-1} |i\rangle \otimes |0\rangle. \quad (2.1)$$

2. Iterate the algorithm for  $\pi/(2\sqrt{2})\sqrt{N/M}$  times by applying the partial diffusion operator  $Y$  on the state  $U_f|\varphi\rangle$  in each iteration, such that it performs the inversion about the mean on a subspace of the system, where

$$Y = (H^{\otimes n} \otimes I)(2|0\rangle\langle 0| - I_{n+1})(H^{\otimes n} \otimes I). \quad (2.2)$$

At any iteration  $q \geq 2$ , the system can be described as follows [6]:

$$U_f|\varphi^q\rangle = a_q \sum_{i=0}^{N-1''} (|i\rangle \otimes |0\rangle) + c_q \sum_{i=0}^{N-1'} (|i\rangle \otimes |0\rangle) + b_q \sum_{i=0}^{N-1'} (|i\rangle \otimes |1\rangle). \quad (2.3)$$

where,

$$a_q = 2\langle\alpha_q\rangle - \alpha_{q-1}, \quad b_q = 2\langle\alpha_q\rangle - c_{q-1}, \quad c_q = -b_{q-1}, \quad (2.4)$$

and

$$\langle\alpha_q\rangle = \left( \left(1 - \frac{M}{N}\right)\alpha_{q-1} + \left(1 - \left(1 - \frac{N}{M}\right)c_{q-1}\right) \right). \quad (2.5)$$

For this algorithm, the success probability is as follows [6]

$$P_s = (1 - \cos(\theta)) \left( \frac{\sin^2((q+1)\theta)}{\sin^2(\theta)} + \frac{\sin^2(q\theta)}{\sin^2(\theta)} \right), \quad (2.6)$$

where  $\cos(\theta) = 1 - M/N$ ,  $0 < \theta \leq \pi/2$ , and the required number of iterations  $q$  is given by:

$$q = \left\lfloor \frac{\pi}{2\theta} \right\rfloor \leq \frac{\pi}{2\sqrt{2}} \sqrt{\frac{N}{M}}. \quad (2.7)$$

The algorithm of Younes *et al.* is noted to be slower than Grover's algorithm [6] for small  $M/N$  by  $\sqrt{2}$ , yet this algorithm is more reliable with high probability than Grover's algorithm and the problem becomes easier with multiple matches.

## 2.2 In Case of Unknown Number of Matches $M$

The previous section describes Younes *et al.*'s algorithm in case of known number of matches  $M$  [6]. However, it is difficult to apply that quantum search algorithm (even Grover's [4] algorithm) without knowing the number of solutions  $M$ , *i.e.* the algorithm is sensitive to the number of iterations which is affected by the number of solutions to the problem itself  $O(\sqrt{N/M})$ . Younes *et al.* described an algorithm that deals with the unknown number of matches  $M$  [6], which is as follows for  $1 \leq M \leq N$ :

1. Start with  $m = 1$  and put  $\lambda = 8/7$ .
2. Choose a positive integer  $j$  uniformly at random such that  $j < m$ .

3. Apply  $j$  iterations of Younes *et al.* on the state:

$$|\varphi\rangle = \frac{1}{\sqrt{N}} \sum_i^n |i\rangle.$$

4. Measure the register assuming its output is  $t$ .

5. If  $f(t)=1$ , then the problem is solved and exit. Otherwise, set  $m$  to the minimum between  $\lambda m$  and  $\sqrt{N}$  and go back to step 2.

Grover's algorithm is employed in ref. [13] to find solutions when  $M$  is unknown, and the estimated solutions grow exponentially when  $M \geq 3N/4$ . Employing Younes *et al.*'s algorithm for finding a solution when the number of solutions is  $M$ , requires  $O(\sqrt{N/M})$  when  $1 \leq M \leq N$  which is better compared to when using Grover's algorithm [13].

### 3 Constructing the Oracle $U_g$

In this section, the oracle  $U_f$  will be used in a way to fit the purpose of finding whether the variable  $x_i$  in the function  $f$  is junta or not before applying the amplification algorithm. We'll provide a way to view and analyze the modification proposed.

#### 3.1 Variable Negation

Any Boolean function  $f$  in positive polarity Reed-Muller form of  $n$  input variables [14], and  $N = 2^n$  can be written as follows:

$$f(x_0, x_1, \dots, x_i, \dots, x_{n-1}) = \bigoplus_{i=0}^{N-1} b_i P_i, \quad (3.1)$$

where

$P_i$  : product term

$$b_i = \begin{cases} 0 : \text{product term does not exist} \\ 1 : \text{product term exists} \end{cases}$$

Let's define  $f_{\bar{x}_i}$  such that,

$$f_{\bar{x}_i} = f(x_0, x_1, \dots, x_{i-1}, \bar{x}_i, x_{i+1}, \dots, x_{n-1}), \quad (3.2)$$

and  $g_{f_{\bar{x}_i}}$  as follows,

$$\begin{aligned}
gff_{\bar{x}_i} &= f \oplus \bar{x}_i \\
&= g(x_0, x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_{n-1}) \\
&= \bigoplus_{j=0}^{K-1} c_j P_j,
\end{aligned} \tag{3.3}$$

where  $K = 2^{n-1}$ ,  $c_j = b_\mu$  and  $\mu$  is the bit representation of the  $j$  term of size  $n$  but only with the bit at position  $i$  equals to 1, *i.e.*  $\mu = j_0 j_1 j_2 \dots j_{i-1} 1 j_{i+1} \dots j_{n-1}$ . It should be noted that  $gff_{\bar{x}_i}$  will decompose the function  $f$  to a lower order general function and the variable  $x_i$  in question will disappear from the definition of  $gff_{\bar{x}_i}$ .

A quantum circuit for the oracle  $U_g$  can be constructed as follows [14]:  $U_g = U_{f_{x_i}} U_f$ , if the variable exists in at least one term in the function  $f$ , and if  $x_i$  is junta then  $U_{f_{x_i}} = U_f$  and then  $U_g = I_n$ , where  $I_n$  is the identity matrix of size  $2^n \times 2^n$ . An illustration of this circuit is shown in fig.2.

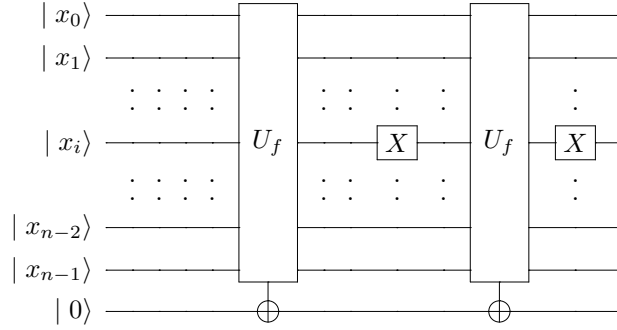


Figure 2: A quantum circuit for the proposed oracle  $U_g$ .

Let's study the case of a general function of 2 inputs which represents all possible 2-variable functions  $f(x_0, x_1)$  in positive polarity Reed-Muller form:

$$f(x_0, x_1) = b_0 \oplus b_1 x_1 \oplus b_2 x_0 \oplus b_3 x_0 x_1. \tag{3.4}$$

Let's assume that the variable in question is  $x_1$ , then

$$f(x_0, \bar{x}_1) = b_0 \oplus b_1 \bar{x}_1 \oplus b_2 x_0 \oplus b_3 x_0 \bar{x}_1. \tag{3.5}$$

It is known that  $\bar{x}_i = x_i \oplus 1$ , so that

$$\begin{aligned}
f(x_0, \bar{x}_1) &= b_0 \oplus b_1(1 \oplus x_1) \oplus b_2x_0 \oplus b_3x_0(1 \oplus x_1) \\
&= b_0 \oplus b_1x_1 \oplus b_1 \oplus b_2x_0 \oplus b_3x_0x_1 \oplus b_3x_0.
\end{aligned} \tag{3.6}$$

Let's define a new function  $g_{ff_{\bar{x}_1}}$ , such that  $g_{ff_{\bar{x}_1}} = f(x_0, x_1) \oplus f(x_0, \bar{x}_1)$ , as follows:

$$\begin{aligned}
g_{ff_{\bar{x}_1}} &= f(x_0, x_1) \oplus f(x_0, \bar{x}_1) \\
&= b_0 \oplus b_1x_1 \oplus b_2x_0 \oplus b_3x_0x_1 \oplus b_0 \oplus b_1x_1 \oplus b_1 \oplus b_2x_0 \oplus b_3x_0x_1 \oplus b_3x_0 \\
&= b_1 \oplus b_3x_0.
\end{aligned} \tag{3.7}$$

Let's study the case of a simple general function of 3 inputs, which represents all possible 3-variable functions,  $f(x_0, x_1, x_2)$  in Reed-Muller form:

$$\begin{aligned}
f(x_0, x_1, x_2) &= b_0 \oplus b_1x_2 \oplus b_2x_1 \oplus b_3x_1x_2 \oplus b_4x_0 \oplus b_5x_0x_2 \\
&\oplus b_6x_0x_1 \oplus b_7x_0x_1x_2.
\end{aligned} \tag{3.8}$$

Let's assume that the variable in question is  $x_0$ , the function  $f(\bar{x}_0, x_1, x_2)$  will be as follows:

$$\begin{aligned}
f(\bar{x}_0, x_1, x_2) &= b_0 \oplus b_1x_2 \oplus b_2x_1 \oplus x_1x_2 \oplus b_4x_0 \oplus b_4 \oplus b_5x_0x_2 \oplus \\
&b_5x_2 \oplus b_6x_0x_1 \oplus b_6x_1 \oplus b_7x_0x_1x_2 \oplus b_7x_1x_2,
\end{aligned} \tag{3.9}$$

Defining  $g_{ff_{\bar{x}_0}}$ , such that  $g_{ff_{\bar{x}_0}} = f(x_0, x_1, x_2) \oplus f(\bar{x}_0, x_1, x_2)$ , will yield

$$g_{ff_{\bar{x}_0}} = b_4 \oplus b_5x_2 \oplus b_6x_1 \oplus b_7x_1x_2. \tag{3.10}$$

It should be noted that the variable  $x_i$  in question disappeared from the general expression of  $g_{ff_{\bar{x}_i}}$ .

## 4 The Proposed Algorithm

Any general Boolean function can be defined in terms of the variable  $x_i$  as follows:

$$f(x_0, x_1, \dots, x_{n-1}) = f_{+x_i} \oplus f_{-x_i}, \tag{4.1}$$

where  $f_{+x_i}$  are the terms in the function  $f$  which contain the variable  $x_i$ , and  $f_{-x_i}$  are the terms in the function  $f$  which do not contain that variable  $x_i$ .

When preparing the function  $g = f \oplus f_{\bar{x}_i}$ , using the new general definition

$$\begin{aligned}
g &= f \oplus f_{\bar{x}_i} \\
g &= (f_{+x_i} \oplus f_{-x_i}) \oplus (f_{+\bar{x}_i} \oplus f_{-\bar{x}_i}) \\
g &= f_{+x_i} \oplus f_{+\bar{x}_i},
\end{aligned} \tag{4.2}$$

where  $f_{+\bar{x}_i}$  are the terms in the function  $f$  which had  $x_i$  and are decomposed to lower order terms not containing  $x_i$ . Then the problem is converted to finding whether  $g$  has at least one solution ( $g \neq 0$ ) or not.

In this section, we will propose the algorithm to test whether a variable  $x_i$  in the Boolean function  $f$  is a junta variable or  $\epsilon$ -far from being a junta variable utilizing the property of the  $U_g$  oracle discussed earlier and Younes *et al.*'s algorithm for unknown number of matches and give basic analysis for the behavior of the algorithm with linear and nonlinear functions in the following section. The steps of the algorithm is as follows:

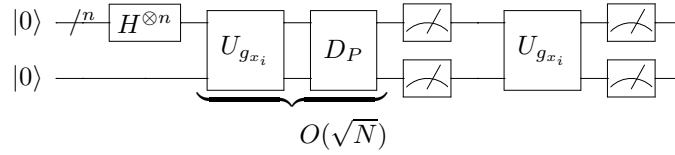


Figure 3: Quantum circuit for the proposed algorithm.

1. Remove any constant terms in  $f$ , if there is any, as follows:
  - (a) Prepare a vector  $v = (0, 0, 0, \dots, 0)$  of size  $n$  such that all qubits are equal to  $|0\rangle$ .
  - (b) If  $f(v) = 1$  which means that the function  $f$  has a constant term, then  $f' = f \oplus 1$ , otherwise  $f' = f$ .
2. Prepare  $g_{x_i} = f' \oplus f'_{\bar{x}_i}$ .
3. Test whether  $g_{x_i}$  has a constant term as follows:
  - (a) Prepare a vector  $v = (0, 0, 0, \dots, 0)$  of size  $n$  such that all qubits are equal to 0.
  - (b) If  $g_{x_i}(v) = 1$ , which means that the vector  $v$  is a solution for the function  $g_{x_i}$  and this implies that  $x_i$  exists in a linear term in the function  $f$ . The variable  $x_i$  will be flagged not junta and then exit.
4. Start with  $m = 1$  and  $\lambda = 8/7$ , where  $\lambda$ , such that  $1 \leq \lambda \leq 4/3$ .
5. Choose a positive integer  $j$  uniformly at random such that  $0 \leq j \leq m - 1$ .
6. Iterate  $j$  iterations of Younes *et al.*'s algorithm on the state:

$$|\varphi\rangle = \frac{1}{\sqrt{N}} \sum_i^n |i\rangle.$$

7. Measure the register assuming its output is  $t_n$ .
8. If  $g_{x_i}(t_n) = 1$ , then  $x_i$  is considered to be not junta and exits.
9. Set  $m$  to the minimum between  $\lambda m$  and  $\sqrt{N}$  and go back to step 5.



## 5 Analysis of The Proposed Algorithm

In this section, we'll discuss the behavior of the proposed algorithm with the suggested oracle modification mentioned in Section 3, assuming of course that the oracle  $U_f$  representing the function  $f$  is a black-box oracle.

### 5.1 Affine Functions

Suppose that the black-box oracle represents an affine Boolean function. An affine Boolean function  $f$  with  $n$  inputs can be generally represented as follows:

$$\begin{aligned} f(x_0, x_1, x_2, \dots, x_{n-1}) &= c_0x_0 \oplus c_1x_1 \oplus \dots \oplus c_{n-1}x_{n-1} \oplus c_n \\ &= \bigoplus_{i=0}^{n-1} c_i x_i \oplus c_n, \end{aligned} \quad (5.1)$$

where the coefficient  $c_i$  decides whether the term that has the variable  $x_i$  exists in the definition of the function  $f$  or not, *i.e.*

$$c_i = \begin{cases} 0, & \text{if } x_i \text{ is junta} \\ 1, & \text{otherwise} \end{cases} \quad (5.2)$$

and  $c_n$  describes generally the affinity of the Boolean function, *i.e.*

$$c_n = \begin{cases} 0, & \text{if the function } f \text{ is not affine} \\ 1, & \text{if the function } f \text{ is affine} \end{cases} \quad (5.3)$$

A linear Boolean function is defined as follows:

$$\begin{aligned} f(x_0, x_1, x_2, \dots, x_{n-1}) &= c_0x_0 \oplus c_1x_1 \oplus \dots \oplus c_{n-1}x_{n-1} \\ &= \bigoplus_{i=0}^{n-1} c_i x_i, \end{aligned} \quad (5.4)$$

where  $c_i$  is as described earlier and  $c_n = 0$ .

If the function  $f$  being tested is an affine Boolean function, *i.e.*  $c_n = 1$ , before proceeding, this constant term must be removed firstly as described in step 1 in the proposed algorithm. This will guarantee that  $g_{x_i}$  will be a linear function with one of the following:

1. If the variable  $x_i$  is not junta, *i.e.*  $c_j = 1$ , the resultant function will be as follows:

$$g_{x_i} = 1, \quad (5.5)$$

which is a constant function that could be easily identified using one evaluation of the function  $g_{x_i}$  as described in the proposed algorithm, *i.e.*  $O(1)$ .

2. If the variable  $x_i$  is junta, *i.e.*  $c_i = 0$ , the resultant function will be as follows:

$$g_{x_i} = 0, \quad (5.6)$$

which is a constant function with no solutions which requires  $O(\sqrt{N})$  oracle calls using the proposed algorithm.

Covering the case when the tested linear function  $f$  is already affine, *i.e.*  $c_n = 1$ , can be handled firstly by removing the constant term and then apply a single evaluation of the function  $g_{x_i}$  as described above.

## 5.2 Nonlinear Functions

Suppose the algorithm is operating on a nonlinear Boolean function  $f$  which is represented as follows:

$$f(x_0, x_1, x_2, \dots, x_{n-1}) = \bigoplus_{i=0}^{2^n-1} c_i t_i, \quad (5.7)$$

where  $t_i = \prod_h x_h$  describes the general form of a single product term, where  $h \leq n$ , and  $c_i$  dictates whether the term  $t_i$  exists or not, then we have the following:

1. If the variable  $x_i$  being tested is a junta variable, the resultant function will be a constant function as in eq.(5.6) and will require  $O(\sqrt{N})$  oracle calls.
2. If the variable  $x_i$  is not a junta variable, it is guaranteed that the function  $g_{x_i}$  will have at least one solution which will be amplified and thus requires  $O(\sqrt{N})$  oracle calls to be found.

## 6 Comparison with Floess's Algorithm

In 2010, Floess introduced a Bernstein-Vazirani-based algorithm for finding which input variables for which the function being tested depend on [9]. The success probability of finding the junta variables for a function has been further amplified using Grover's search algorithm [4].

**Single term Boolean function of order  $m$  amplification:** A drawback of Floess algorithm appears when the function being tested is a single product term Boolean function of  $m$  variables. If the Boolean function being tested is a product of  $m$  input variables such that  $m \leq n$ , which should be an unknown fact about the function being tested, as follows:

$$f(x_0, x_1, \dots, x_{n-1}) = \prod_{j=1}^m x_j, \quad (6.1)$$

then the number of iterations required for Grover’s algorithm must be estimated as Grover’s algorithm is sensitive to number of iterations [11, 13]. Floess shows that for a product of  $m$  input variables, the required number of iterations for Grover’s algorithm can be estimated using a circuit which requires  $O(2^m)$  oracle calls [15], which doesn’t provide any speedup compared to a classical algorithm counterpart. The proposed algorithm, however, introduces a quadratic speedup compared to Floess’s algorithm, *i.e.*  $O(\sqrt{2^m})$ .

**Multiple terms Boolean function:** Whether the decomposed function  $g$  is of single term or several terms which implies several solutions, the expected function calls of the proposed algorithm is  $O(\sqrt{N/M})$ , when  $M$  is  $1 \leq M \leq N$ . However, Floess didn’t cover the case when the unknown function  $f$  is composed of multiple terms with different degrees.

## 7 Conclusion

The paper proposed a quantum algorithm to test if a certain variable of a given Boolean function  $f$  with  $n$  variables is junta or  $\epsilon$ -far from being junta. The Boolean function is assumed to be provided as a black-box. The black-box Boolean function is used to construct another black-box using two copies of the given black-box. It was shown that the constructed black-box will have no solutions if the variable in question is junta and has at least one solution if the variable in question is not junta. The number of solutions of the constructed black-box is assumed to be unknown where an amplitude amplification technique is used that marks the solutions with entanglement with partial diffusion operator to find whether the constructed black-box has at least one solution or does not have any solutions using  $O(\sqrt{2^n})$  queries to the constructed black-box.

It was shown that the proposed algorithm can handle any Boolean function provided as a black-box without any restrictions on the structure of the Boolean function, where the relevant work proposed by others that tests certain classes of Boolean functions can be considered as special cases of the proposed algorithm.

## References

- [1] A. Blum, "Relevant examples and relevant features: thoughts from computational learning theory," *In AAAI Fall Symp. on 'Relevance'*, pp. 18-22, 1994.
- [2] A. Blum and P. Langley, "Selection of relevant features and examples in machine learning," *Artificial Intelligence*, vol. 97, no. 1-2, pp. 245-271, 1997.
- [3] P. Shor, "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer," *SIAM Review*, vol. 41, no. 2, pp. 303-332, 1999.

- [4] L. Grover, "A fast quantum mechanical algorithm for database search," in *STOC 96 Proc. of the 28th annual ACM Syem. on Theory of computing*, 1996, pp. 212-219.
- [5] G. Brassard, P. Høyer, M. Mosca, and A. Tapp, "Quantum amplitude amplification and estimation". Available: arXiv:quant-ph/0005055.
- [6] [5]A. Younes, J. Rowe and J. Miller, "Enhanced quantum searching via entanglement and partial diffusion," *Physica D: Nonlinear Phenomena*, vol. 237, no. 8, pp. 1074-1078, 2008.
- [7] E. Bernstein and U. Vazirani, "Quantum Complexity Theory," *SIAM Journal on Computing*, vol. 26, no. 5, pp. 1411-1473, 1997.
- [8] A. Atc and R. Servedio, "Quantum Algorithms for Learning and Testing Juntas," *Quantum Information Processing*, vol. 6, no. 5, pp. 323-348, 2007.
- [9] D. Floess, E. Anderson and M. Hillary, "Quantum algorithms for testing and learning Boolean functions," *Mathematical Structures in Computer Science*, vol. 23, no. 02, pp. 386-398, 2013.
- [10] H. Li and L. Yang, "A quantum algorithm for approximating the influences of Boolean functions and its applications," *Quantum Information Processing*, vol. 14, no. 6, pp. 1787-1797, 2015.
- [11] A. Younes, "Strength and weakness in Grover's quantum search algorithm". Available: arXiv:0811.4481v1
- [12] G. Brassard, P. Høyer, and A. Tapp, Quantum counting. Available: arXiv:quant-ph/9805082
- [13] M. Boyer, G. Brassard, P. Høyer and A. Tapp, "Tight Bounds on Quantum Searching," *Fortschritte der Physik*, vol. 46, no. 4-5, pp. 493-505, 1998.
- [14] A. Younes and J. Miller, "Representation of Boolean quantum circuits as Reed-Muller expansions," *International Journal of Electronics*, vol. 91, no. 7, pp. 431-444, 2004.
- [15] D. Floess, "Quantum mechanics and Boolean functions," unpublished M.S. thesis, Heriot-Watt University.